NAMA : MUSYTAHAR SYAH

NIM : 1117000441

MATKUL : DESIGN ANALYSIS ALGORITHM

TUGAS 1 DAA

# Persiapan

Sebelum mengerjakan tugas ini Anda diwajibkan untuk membuat akun di **GitHub.com** kemudian melakukan *forking* terhadap repositori MK-DAA milik dosen pengampu. Jangan lupa meng-*install* GitHub Desktop di komputer Anda untuk memudahkan proses *forking* maupun *pull request* nantinya.

# Pengantar

Bilangan Fibonacci adalah suatu deret dengan bentuk sebagai berikut:

Pembentukan pola bilangan Fibonacci didasarkan pada aturan berikut:

* Bilangan Fibonacci ke-0 adalah 0
* Bilangan Fibonacci ke-1 adalah 1
* Bilangan Fibonacci ke-*n* adalah bilangan Fibonacci ke-(*n*-1) dijumlahkan dengan bilangan Fibonacci ke-(*n*-2)

atau dalam rumus matematika dituliskan sebagai berikut:

Dengan memanfaatkan algoritma rekursif, contoh kode program untuk mendapatkan bilangan Fibonacci ke-*n* dalam bahasa C# adalah sebagai berikut:

class Fibonacci

**{**

**public** double get**(**int n**)**

**{**

**if** **(**n **<=** 1**)** **return** n**;**

**else** **return** get**(**n **-** 1**)** **+** get**(**n **-** 2**);**

**}**

**}**

Dan untuk mengujinya bisa digunakan *main program* berikut:

class Program

**{**

static void Main**(**string**[]** args**)**

**{**

int n**;**

double result**;**

Fibonacci fibo **=** **new** Fibonacci**();**

Console**.**Write**(**"Request bilangan Fibonacci ke : "**);**

n **=** Int32**.**Parse**(**Console**.**ReadLine**());**

result **=** fibo**.**get**(**n**);**

Console**.**WriteLine**(**"Bilangan Fibonacci ke-{0} adalah {1}"**,**n**,**result**);**

Console**.**Read**();**

**}**

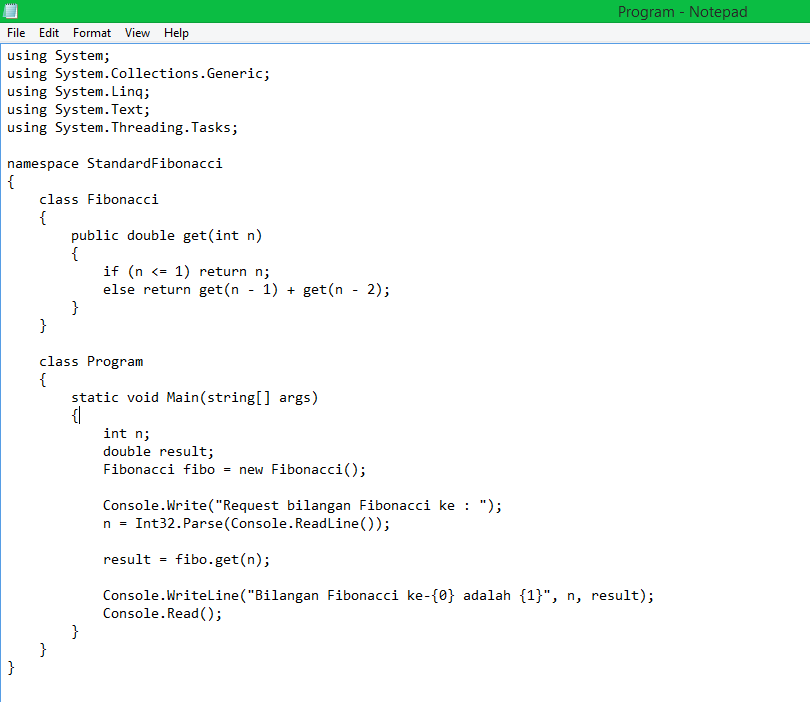
**}**

Anda bisa menyalin kode program di atas pada repositori berikut: <https://github.com/ksatria/MK-DAA/tree/master/Code/StandardFibonacci>

# Percobaan

Cobalah untuk membuat atau menggunakan program di atas untuk mendapatkan bilangan Fibonacci ke-0, 1, 2, 10, 50, 100, dan seterusnya. Apakah waktu yang dibutuhkan untuk mendapatkan masing-masing bilangan Fibonacci tersebut tetap sama

**Jawab:**

Source code yang saya gunakan sama dengan *code* StandartFibonacci sebagai berikut 

Dari hasil yang telah saya lakukan dengan menggunakan program mendapatkan hasil seperti berikut ini

1. Mendapatkan bilangan *Fibonacci* ke-0

Hasil *compile* program mencari *Fibonacci* ke-0 sebagai berikut



Penjelasan :

Hasil *request* bilangan *Fibonacci* ke 0 adalah 0 dengan waktu pencarian hasil bilangan sangat cepat sekali.

1. Mendapatkan bilangan *Fibonacci* ke-1

Hasil *compile* program mencari *Fibonacc*i ke-1 sebagai berikut



Penjelasan :

Hasil *request* bilangan *Fibonacci* ke 1 adalah 1 dengan waktu pencarian hasil bilangan juga sangat cepat sekali.

1. Mendapatkan bilangan *Fibonacci* ke-2

Hasil *compile* program mencari *Fibonacci* ke-2 sebagai berikut



Penjelasan :

Hasil *request* bilangan *Fibonacci* ke 2 adalah 1 dengan waktu pencarian hasil bilangan juga sangat cepat sekali. Dan bilangan *Fibonacci* ke-2 sama dengan Fibonacci ke-1

1. Mendapatkan bilangan *Fibonacci* ke-10

Hasil *compile* program mencari *Fibonacci* ke-10 sebagai berikut

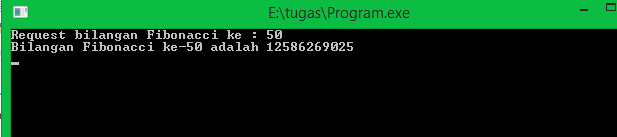


Penjelasan :

Hasil *request* bilangan *Fibonacci* ke-10 adalah 55 dengan waktu pencarian hasil bilangan juga cepat .

1. Mendapatkan bilangan *Fibonacci* ke-50

Hasil *compile* program mencari *Fibonacci* ke-50 sebagai berikut

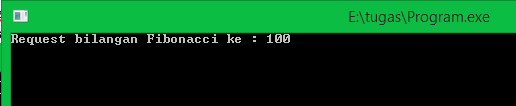


Penjelasan :

Hasil *request* bilangan *Fibonacci* ke-50 adalah 12586269025 dengan waktu pencarian hasil bilangan yang lambat .

1. Mendapatkan bilangan Fibonacci ke-100

Hasil *compile* program mencari *Fibonacci* ke-100 sebagai berikut



Penjelasan :

Hasil *reques*t bilangan *Fibonacci* ke-100 adalah belum saya temukan bahkan saya menunggu berjam-jam belum mendapatkan hasil nya.

Kesimpulan percobaan:

Pada proses mendapatkan bilangan pada *Fibonacci* ke-0,1,2,10,50 dan 100 saya menemukan bahwa waktu yang digunakan untuk pencarian bilangan berbeda beda dan saya temmukan kendala waktu dimulai dari encarian bilangan *Fibonacci* yang ke-50 daan saya tidak menemukan hasil bilangan dari *Fibonacci* ke-100 munkin dikarenakan proses *looping* yang cukup lama pada *source code* diatas.

# Tugas

Sederhanakan algoritma untuk mendapatkan bilangan Fibonacci ke-*n* sehingga waktu yang dibutuhkan untuk mendapatkan masing-masing bilangan relatif sama!

**Jawab**

Pada percobaan diatas memang masih memakan waktu lama dalam pemanggilan data nya dengan menggunakan *rekursif*. Akan tetapi nilai bilangan FIBONACCI suku ke-n dalam deret dapat dihitung dengan menggunakan metode lain yaitu menggunakan *iterasi* .

Dalam teknik iterasi nilai suku bilangan FIBONACCI dihitung dengan melaksanakan suatu proses perulangan yang memanfaatkan nilai bilangan FIBONACCI suku ke-1 sebagai dasar perhitungan untuk mencari nilai pada suku ke-2.Nilai pada suku ke-2 digunakan sebagai dasar mencari nilaisuku ke-3. Nilai pada suku ke-3 digunakan sebagai dasar mencari nilai suku ke-4. Demikian seterusnya hingga nilai bilangan FIBONACCI pada suku ke-n yang dikehendaki ditemukan

Algoritma untuk mencari nilai bilangan FIBONACCI suku ke-n dengan menerapkan teknik iterasi adalah dituliskan sebagai berikut ini : Masukkan harga suku bilangan FIBONACCI yang akan akan dicari (n).FIBONACCI merupakan variabel untuk menyimpan harga bilangan Fibonacci yang dicari.BANTU merupakan variabel bantuan untuk menampung hasil perhitungan sementara.

Algoritma iterasi:

1. Mulai
2. Tentukan harga-harga awal I, FIBONACCI, dan TERAKHIR

I = 1

FIBONACCI = 1

TERAKHIR = 0

1. Cek harga n

IF n = 1

Jika ya, maka FIBONACCI = 1, lanjutkan ke langkah-6

1. Proses berulang langkah-4 hingga langkah-5 untuk menghitung FIBONACCI

Hitung

BANTU = FIBONACCI

I = I + 1

1. Hitung

FIBONACCI = FIBONACCI + TERAKHIR

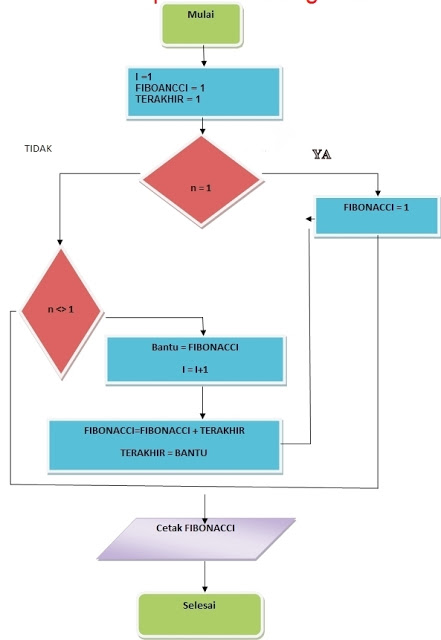
TERAKHIR = BANTU

1. Cetak Hasil

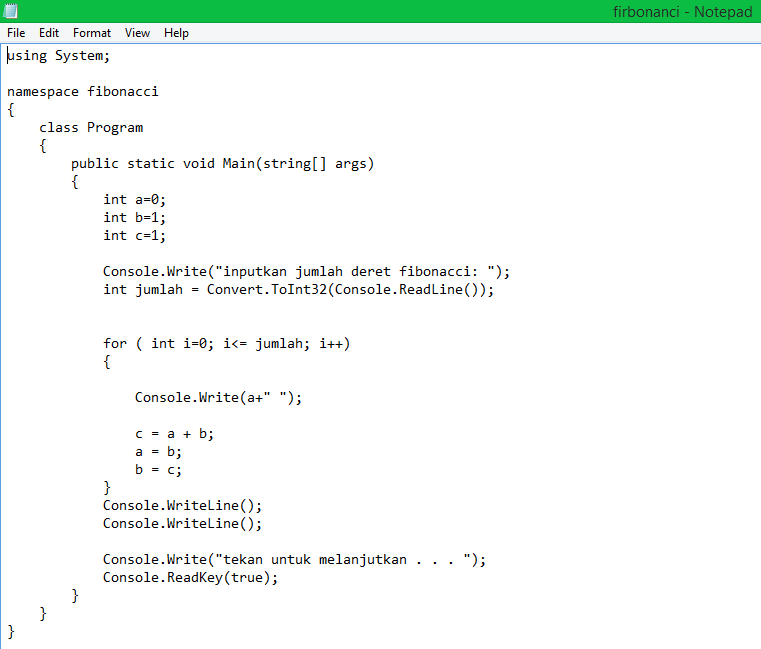
FIBONACCI

1. Selesai

Flowchart iterasi Fibonacci



Source code iterasi Fibonacci



Hasil percobaan source kode iterasi Fibonacci

1. Deret 0



1. Deret 1



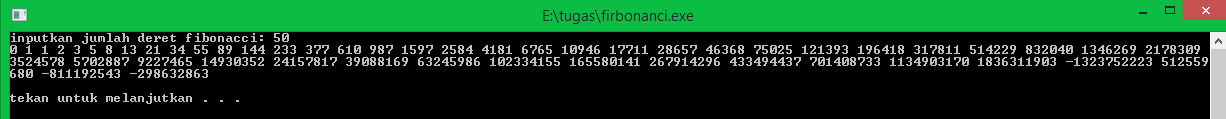
1. Deret 2



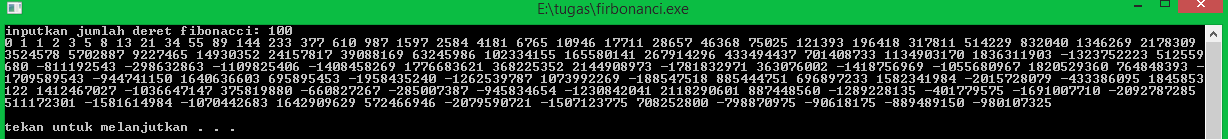
1. Deret 10



1. Deret 50



1. Deret 100



# Kesimpulan

Dalam contoh kasus perhitungan nilai bilangan FIBONACCI ini, secara umum teknik iterasi mampu melaksanakan proses perhitungan secara lebih efisien dibandingkan jika diselesaikan dengan menerapkan teknik rekursi. Dan tentu saja, hasil perhitungannya akan dapat ditemukan secara lebih cepat. Untuk jenis kasus ini, penggunaan recursive sangat tidak disarankan, karena akan berakibat kompleksitas komputasi meningkat dan waktu eksekusinya juga meningkat.